Remarks:

Reconsideration of the above referenced application in view of the enclosed amendments and discussion and remarks is requested. Claims 1, 14 and 20 are amended. Claim 31 is added to recite a further embodiment of the invention as described in the specification as filed. Claims 1 to 31 are pending in the application.

ARGUMENT

Claims 1-30 are rejected under 35 U.S.C. § 103(a) as being unpatentable over U.S. Patent No. 6,081,890 to Datta (hereinafter, "Datta"), in view of *Extensible Firmware Specification Version 1.02* by Intel Corp. (hereinafter, "EFI Spec."). This rejection is respectfully traversed, and Claims 1-30 are believed allowable based on the following discussion.

The Examiner misapplies the teachings in the EFI Spec. to the techniques taught by Datta. The EFI Spec. describes an interface between the operating system (OS) and the platform firmware. The interface is in the form of data tables that contain platform-related information, and boot and runtime service calls that are available to the OS and its loader. Together, these provide a standard environment for booting an OS. The EFI specification is designed as a pure interface specification. As such, the specification defines the set of interfaces and structures that platform firmware must implement. Similarly, the specification defines the set of interfaces and structures that the OS may use in booting. How either the firmware developer chooses to implement the required elements or the OS developer chooses to make use of those interfaces and structures is an implementation decision left for the developer. [Introduction, EFI Spec. page 1]

Implementation of the interface as applied to booting a computer is not taught. Applying the specification itself to the teaching of Datta will not result in Applicants' claimed invention. Datta teaches a system for use on an Itanium® processor calling down into IA32 firmware, namely to mix modules from different instruction sets. Datta teaches a system for new firmware on an Itanium® processor to leverage the PC/AT BIOS written in x86 real-mode code. There is no motivation to combine this teaching with an EFI architecture, as Datta solves a different

problem. Datta teaches a method to use legacy and native firmware modules written for legacy and native ISAs. Datta teaches a method to use a Prologue routine to link two firmware modules. Datta does not teach a method to bootstrap a system having an EFI architecture to sequence and access architecturally defined firmware modules.

The EFI Spec. more specifically describes driver modules. Its purpose is not to define the sequencing and invocation of modules. It is improper to combine the teachings of Datta and the EFI Spec. Moreover, doing so will not result in Applicants' invention. Combining Datta with the literal EFI Spec. would be like mixing Itanium® drivers with BIOS/x86 drivers – apples with oranges. One of ordinary skill in the art would not do this. One might mix a native instruction set driver with EFI Byte Code (EBC), which is like translating to another instruction set architecture (ISA). But this does not result in Applicants' invention.

The claimed invention is directed toward a model for having a specific file type for reset, or restart, events and then an agency to sequence a series of possible heterogeneous modules/files thereafter. Some claimed embodiments enable authentication of the modules.

Specifically addressing the Examiner's cited references, the EFI Spec. pages 323-324 define a number of globally defined variables. The specification does not define exactly how the variables are to be used in initializing a boot routine. The Boot variables do not fully describe the boot process, but enable the programmer to implement a structurally defined system. EFI Spec. page 319 describes only that a boot order list is maintained in a globally defined NVRAM variable. At no time does the EFI Spec. teach or suggest that the hardware will automatically execute a volume top file at a specified location in firmware in response to a reset event. (See the specification, at least at page 3, last paragraph and page 6, last paragraph. Nor does Datta teach or suggest that a volume top file be architecturally defined and automatically run from a specific location upon a reset event. In fact, Datta teaches away from the claimed invention. Datta teaches that an entry point – not a volume top file – be executed from a location at 16 bytes below 4 GB, which is defined by processor. However, this location differs based on the hardware and firmware developers must get this entry point location from the hardware spec and code the firmware accordingly.

In contrast, Applicants' claimed invention requires an architecturally defined VTF, not just a mere entry point, is automatically transferred to by the hardware upon a reset event,

circumventing software defined entry point locations, as taught by Datta. Specifically, the Examiner cites Datta, Col. 6, lines 27-35, as teaching a volume top file. Datta describes that an entry point of the firmware is located at 16 bytes below 4GB. This entry point is not the same as the volume top file as recited in Applicants' claims. Datta's entry point is defined by software and not transferred to directly by the hardware. Further, Datta does not teach that the entry point has an architecturally defined volume top file which executes automatically by the hardware in response to a restart event. Therefore, all claims are believed allowable and should be allowed to issue at the earliest possible time.

CONCLUSION

In view of the foregoing, Claims 1-31 are all in condition for allowance. If the Examiner has any questions, the Examiner is invited to contact the undersigned at (703) 633-6845. Early issuance of Notice of Allowance is respectfully requested. Please charge any shortage of fees in connection with the filing of this paper, including extension of time fees, to Deposit Account 02-2666 and please credit any excess fees to such account.

Respectfully submitted,

Dated: Jul. 13, 2006

s/Joni D. Stutman-Horn/

Joni D. Stutman-Horn, Reg. No. 42,173
Patent Attorney
Intel Corporation
(703) 633-6845

c/o Blakely, Sokoloff, Taylor &
Zafman, LLP
12400 Wilshire Blvd.
Seventh Floor
Los Angeles, CA 90025-1026